続

実設計に応用できる演算回路スキルを身につけよう

デバイスの記事

外村元伸

第1回 ディジタル画像処理のための各種補間法を理解する

本連載では，ディジタル画像処理機器で利用される画像の拡大および縮小スケーリングを題材に，高性能な演算回路の設計方法を解説する．今回は，ディジタル画像処理で用いられる「アルファ・ブレンディング」や画像拡大・縮小スケーリングのための各種補間法の基礎を取り上げる．なお，本連載は本誌2006年5月号，pp.39-48，「特集1 第2章 LSIの中の基本部品を原理から理解する」で紹介した2進数や演算器の基本回路構成，線形補間の考えかたをベースにして話を進める．

（編集部）

本誌2006年5月号，pp.39-48，「LSIの中の基本部品を原理から理解する」では，線形補間という身の回りの製品でも利用されている演算の専用回路化を題材に，演算器とその回路設計の基礎について解説しました．本連載は，その具体的な応用例として，「画像処理の演算器設計」を取り上げます．画像処理を選んだ理由は，表示された画像を見れば自分の作った回路のよしあしが一目瞭然だからです．また，最近はいろいろな分野でディジタル画像処理機器が普及しています．みなさんの中にもこれらの設計に携わっている，あるいは今後携わっていく方も少なくないでしょう．本稿がみなさんのおしごとに役立つことを期待しています．

1. 画像合成「アルファ・ブレンディング」

二つの画面AとBを割合α($0 \leqq \alpha \leqq 1$)で合成表示することを考えた場合，これは線形補間です．割合αにちなんで，画面合成という行為は「アルファ・ブレンディング」と呼ばれています．ここでは，アルファ・ブレンディングの画面合成に線形補間演算器を応用した特徴的な実現法を紹介します．

● 循環2進小数なので簡単化できる

ディジタル・テレビの規格では，合成の割合αは8ビットの整数（10進で0～255）で定義しています．これがαレジスタに格納されているとします．αレジスタの内容が0と255のとき，それぞれAとBの一方の画面のみを表示するようになっています．

αレジスタの内容から$0 \leqq \alpha \leqq 1$の表現に変換するためには，**図1**に示すように1/255×αレジスタの計算を行わなければなりません．普通に考えれば，これには乗算が必要になります．

実際には，1/255の定数乗算を行うのを避けて，近似的に1/256としてαレジスタの内容の単純な固定小数点位置のシフトに置き換えています．そして，比較器でαレジスタを判定し，レジスタの内容が255であれば画面Bのみを表示するように設計しているようです．

ところで2006年5月号，pp.39-48の記事の中で，循環2進小数の例として，

$$1/255 = [0.0000000100000001\cdots]\mathrm{b} = [0.\dot{0}\dot{0}\dot{0}\dot{0}\dot{0}\dot{0}\dot{0}\dot{1}]\mathrm{b}$$

を挙げました．αレジスタ＝$[d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0]$bとすれば，1/255×$\alpha$レジスタ＝$[.d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0 d_7 d_6 d_5 d_4 d_3 d_2 d_1 d_0]$bと単純に表現できるので，適当なビット位置$i$で$d_i = 1$ならば切り上げるだけでよいのです．任

KeyWord
演算回路，画像合成，アルファ・ブレンディング，線形補間，最近傍選択法，バイリニア法，バイキュービック補間法，コンボリューション法，スプライン法

意のビット長nの場合も次の式(1)が成り立つので，循環2進小数になっています．

$$\frac{1}{\sum_{i=0}^{n-1} 2^i} = \frac{1}{2^n - 1} = \sum_{j=1}^{\infty} 2^{-jn} = 2^{-n} + 2^{-2n} + \cdots \quad \cdots\cdots\cdots\cdots(1)$$

2. 画像拡大・縮小スケーリング法いろいろ

次に，画像の拡大・縮小スケーリング表示へ線形補間演算を応用してみましょう．実際の応用場面では，線形(1次)補間では不十分で，2次補間，3次補間が必要な場合が少なからずあります．ここでは，3乗演算を含む3次補間演算器の設計法について詳しく説明します．こう言うと，「3乗を含む演算は複雑なためコストがかかる．直接演算はあきらめて，演算結果のテーブルを利用して設計しよう」とたいていの方は考えるのではないでしょうか．

しかし，線形補間演算器の原理や基礎を押さえていれば，ちょっとしたくふうで意外と簡単に，しかもコンパクトに直接演算を用いて実現できてしまいます．従来のみなさんの常識を変えたいというのが本稿のねらいです．またここで解説するテクニックは，画像処理以外でも，例えば音声処理や通信処理など，いろいろな分野に応用できます．

● 表示画面の違う機器でデータを共有するとき必要な技術

ひと口にディジタル画像処理と言っても，民生機器(テレビ，パソコン，ゲーム機，カメラ，プリンタなど)，携帯電話，車載機器，医療機器など，応用分野はさまざまです．

これらの機器には，それぞれ入力用のカメラと出力用のディスプレイが必要に応じて付いています．また，業務用と個人用，机上用と携帯用，大型と小型，動画用と静止画用，高解像度と低解像度などという分類もあります．これら各種ディジタル画像処理機器で取得した画像データを互いに転送して共用しようとするとき，機器の表示画面に応じた画像の拡大・縮小変換が要求されます．これを画像の拡大・縮小スケーリングと呼んでいます．

この場合，適当な原画像の補間計算によって表示位置の画素値を求める必要があります．とくに，つぶれないように縮小すること，滑らかに拡大することが要求されるため，提案されている各種補間方式の中から，画質に応じて適切な方式を選択しなければなりません．

● 直接演算すればコンパクトで高速なスケーリングが可能

さまざまな状況の画質で画像が扱われるため，画像の拡大・縮小スケーリングのための補間計算法の種類が多様化し，複雑化してきています．それらに伴う回路規模や消費電力の増大はできるかぎり抑えなければなりません．

現状ではこうした拡大・縮小スケーリングについて最適な演算器設計がなされているとは思えません．3次多項式の計算は複雑だという考えから，規模が大きいにもかかわらず演算結果のテーブルを用いて実現しているからです．テーブルというメモリを使うことで，メモリの参照時間が余計にかかります．これを直接演算することにより処理時

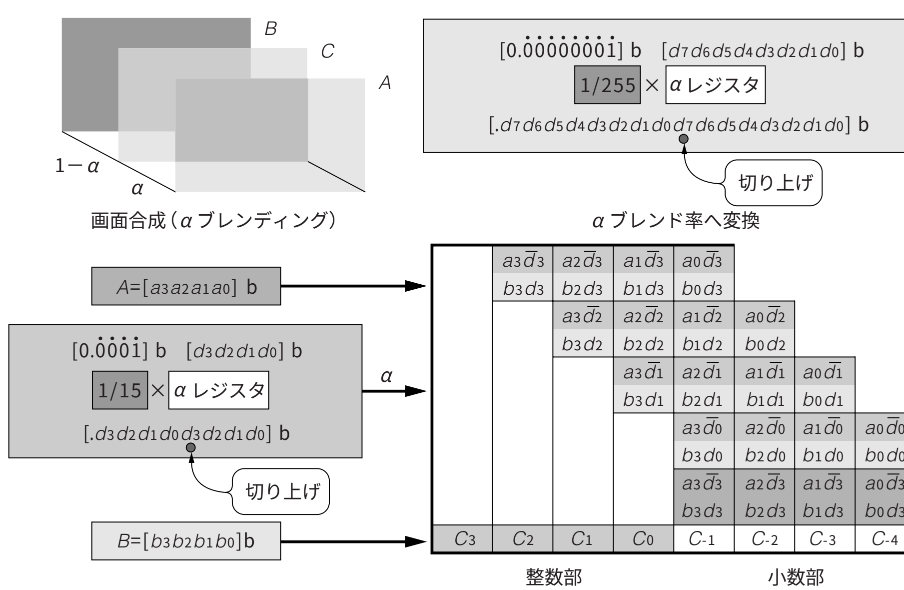


図1
アルファ・ブレンディング画面合成へ線形補間演算器を応用

積項の積み上げ表現形式については，誌面のつごうで4ビットの例(1/15)×(αレジスタ)を示す．d_3による切り上げの項が余分に加わる

間を短縮できます．

また，直接演算することで，演算精度も容易に保証できます．そのため，演算幅のサイズを最小化でき，演算器をコンパクトにすることができます．

● 補間精度を上げるにはバイキュービック補間法を用いる

画像の拡大・縮小スケーリングによく使われるのは，最近傍選択（nearest-neighbor）法とバイリニア（bilinear）補間法です．

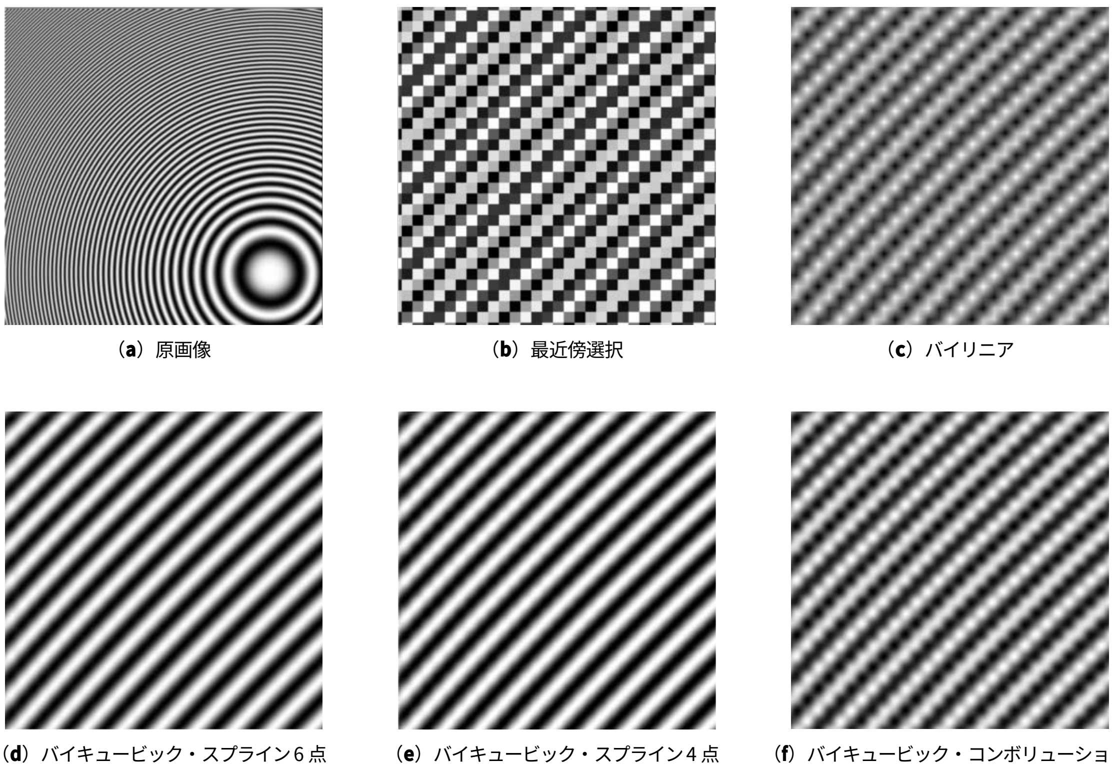


図2　各種補間法に基づく画像拡大スケーリング表示結果の比較

各辺の拡大率が10倍（面積拡大率100倍）の場合の表示比較を示す

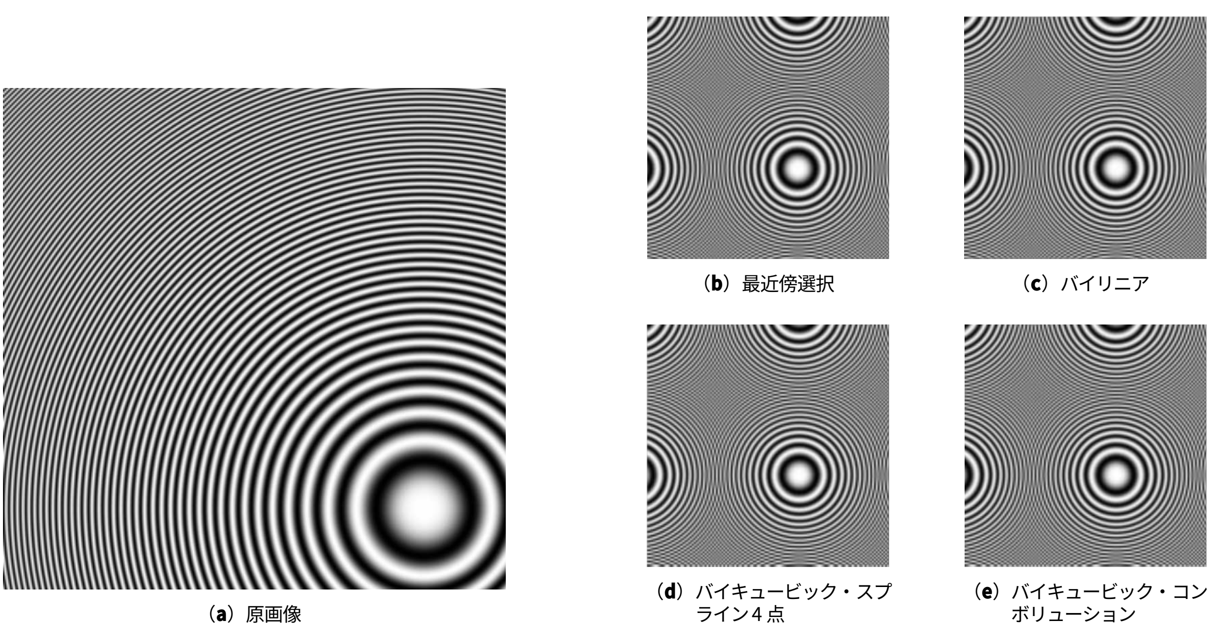


図3　各種補間法に基づく画像縮小スケーリング表示結果の比較

各辺の縮小率が1/3倍（面積縮小率1/9倍）の場合の表示比較を示す．拡大の場合と比べて補間方式による違いがはっきりとわからない

最近傍選択法は，表示位置にもっとも近い原画素を選択します．したがって，拡大の場合は補間領域が平たんな表示になります．逆に縮小の場合，原画素を間引くかたちになるため，適度な縮小率のときは，わりと良好な表示になります．

バイリニア補間法は，水平方向2個×垂直方向2個の合計4個の原画素を端点として線形補間します．バイリニアの「バイ」は，水平と垂直の両方向という意味です．適度な拡大率では，端点画素の位置からの割合で線形補間して表示点における画素値を求めるため，良好な表示結果となります．しかし，過度の拡大率になると，ボケたような表示結果となります．縮小の場合は，線形補間の画素値が最近傍の画素値に近く，最近傍選択法との違いがほとんど出ません．そして，もっと高精度な補間法を採用しても，縮小では補間誤差が小さいため，最近傍選択法と変わらないような表示結果になります．したがって，補間精度が問題になるのは拡大の場合ということになります．

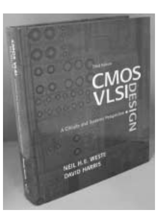
補間精度を改善するため，原画素4×4個を用いるバイキュービック（bicubic）補間法が使われています．バイキュービック補間には，コンボリューション（convolution）法とスプライン（spline）法があります．両者の違いについてはあまりよく知られていないばかりか，混乱さえあります．また，スプライン法は，参照画素数が4×4個だけでなく，任意の参照画素数に対しても適用可能です．例えば $n \geqq 4$ に対して原画素 $n \times n$ 個が考えられます．

図2に，各種補間法に基づく画像拡大スケーリング表示結果を比較したものを示します．水平方向および垂直方向の拡大率はともに10倍です．したがって，面積当たりの拡大率は100倍です．**図3**には画像縮小スケーリング表示結果を比較したものを示します．水平方向および垂直方向の

COLUMN：お勧めの演算回路のテキスト

演算回路設計の参考書として最近発行された2冊の書籍を紹介します．トランジスタ・レベルで演算回路の設計原理を詳しく知りたい方にはWesteの著書を，HDL記述レベルで設計したいのでトランジスタ・レベルのことはとくに気にしたくない方にはDeschampsの著書がよいと思います．両著書とも，演算回路に関する和書ではまだ扱っていない最新の設計技術を解説しています．

CMOS VLSI Design：A Circuits and Systems Perspective（3rd Edition）

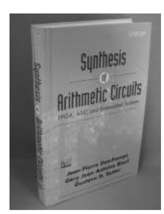


Neil H. E. Weste，David Harris 著
Addison-Wesley 刊
ISBN：0321149017
24×21cm　967ページ
14,578円（税込み．Amazon.co.jpの2006年5月23日現在の価格）
2004年5月（第3版）

本書は，2004年で3rd editionになりました．1st edition（1985年）と2nd edition（1993年）は，Kamran Eshraghianとの共著でした．1st editionのころはCMOS回路に関する参考図書が少なかったので，筆者にとっては貴重な本でした．その後，時代の流れに沿うように，2nd editionは大幅に改訂されました．そして，3rd editionでは共著者がDavid Harrisに交代し，まったく新しい内容に生まれ変わりました．

とくに，さまざまな回路方式に対して特徴を分析し，比較しています．パス・トランジスタ回路についてもていねいに説明しています．演算回路についても，最近の考えかたを取り入れています．付録では，Verilog HDLとVHDLの両方の記述について説明しています．

Synthesis of Arithmetic Circuits, FPGA, ASIC and Embedded Systems



Jean-Pierre Deschamps，Géry Jean Antoine Bioul，Gustavo D. Sutter 著
Wiley-Interscience 刊
ISBN：0471687839
24×16cm　556ページ
13,370円（税込み．Amazon.co.jpの2006年5月23日現在の価格）
2006年3月（初版第1版）

演算回路に興味のある方ならぜひ一読してみたいと思いたくなるようなタイトルです．しかし，内容はWesteの著作のようなトランジスタ・レベルの説明はまったくなく，ゲート論理以上のレベルを扱っています．とくに，HDL記述のレベルで扱うことを想定した内容になっています．残念なのは，VHDL記述のみが紹介されていることです．Verilog HDL記述については，リンクされているホームページからもダウンロードできません．

本書の大きな特徴は，数表現について詳しく取り上げていることです．また，過去に使われてきた方式から最新の方式まで扱っているので，演算回路をHDL記述のレベルで勉強したいみなさんには良い参考書になると思います．暗号やエラー検出，エラー訂正処理では，有限体（ガロア体），すなわち排他的論理和の数学がよく使われますが，有限体の演算回路もきちんと説明しています．浮動小数点演算についても要点をまとめてあります．

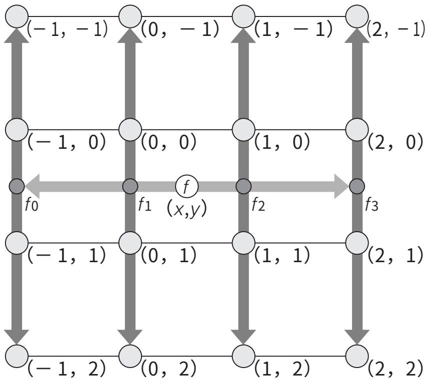


図4　バイキュービック補間のために垂直・水平方向に分解して3次補間を適用

2次元のままで補間計算を扱うと複雑な式になる．垂直方向と水平方向に1次元分解することで，これから説明する1次元のキュービック補間方式の単純な式の繰り返しにより2次元化できる

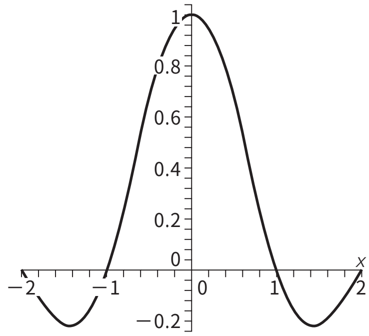


図5　sinc(x)関数

範囲は，x＝[−2，2]

縮小率はともに1/3倍なので，面積当たりの縮小率は1/9倍です．**図2**と**図3**に使用したのは，ゾーンプレートと呼ばれる人工的に描画生成した画像で，画質の評価によく使われます．描画法に関する詳細は次回以降の記事の中で説明します．

● 従来のキュービック補間法

本題のキュービック補間法について説明する前に，まず基本的なことを押さえておきましょう．

4点x_{-1}，x_0，x_1，x_2を通る3次曲線$f(x)$は，式(2)のように表され，一意に決定されます．ここで，$f(x_{-1})=f_{-1}$，$f(x_0)=f_0$，$f(x_1)=f_1$，$f(x_2)=f_2$であることは明らかです．

$$
\begin{aligned}
f(x)=&\frac{(x-x_0)(x-x_1)(x-x_2)}{(x_{-1}-x_0)(x_{-1}-x_1)(x_{-1}-x_2)}f_{-1}\\
&+\frac{(x-x_{-1})(x-x_1)(x-x_2)}{(x_0-x_{-1})(x_0-x_1)(x_0-x_2)}f_0\\
&+\frac{(x-x_{-1})(x-x_0)(x-x_2)}{(x_1-x_{-1})(x_1-x_0)(x_1-x_2)}f_1\\
&+\frac{(x-x_{-1})(x-x_0)(x-x_1)}{(x_2-x_{-1})(x_2-x_0)(x_2-x_1)}f_2
\end{aligned}
\quad\cdots\cdots\cdots\cdots\cdots(2)
$$

式(2)は，キュービック・ラグランジェ(cubic Lagrange)補間式と呼ばれています．**図4**に，2次元の画像を垂直方向と水平方向の1次元に分解して，2度に分けてキュービック・ラグランジェ補間を適用して行うようすを示します．このように，キュービック・ラグランジェ補間式は4点を厳密に通る3次曲線なので，このままでは誤差を伴うような観測値の曲線当てはめ(curve fitting)には使いにくいことになります．

そこで，一般化された補間多項式によって3次の補間法を導くために，以下ではよく使われているキュービック・コンボリューション補間法について説明します．

アナログの世界では，**図5**に示す$\mathrm{sinc}(x)=\sin\pi x/\pi x$が理想的なフィルタと言われています．しかし，ディジタルの世界では数値的に収束が遅いので，その3次近似式として考えだされたものがキュービック・コンボリューション法です．

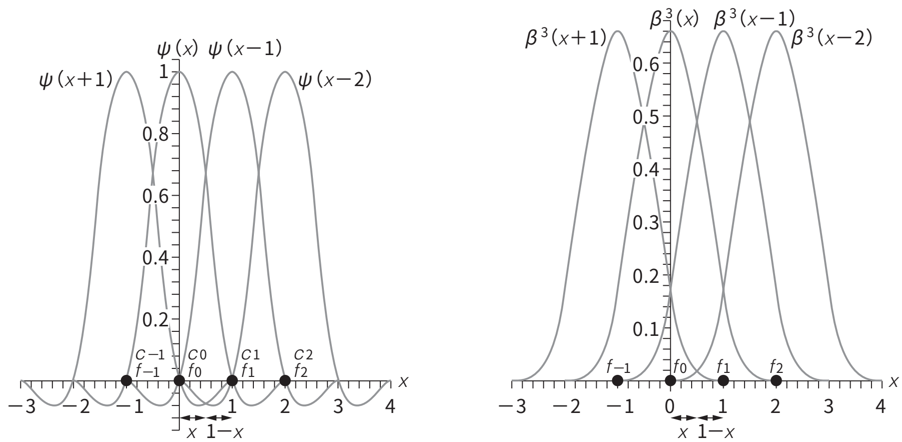
式(3)のような二つの3次曲線をベースに接続した重み関数$\varphi(\mathrm{x})$を考えます．

$$
\varphi(x)=\begin{cases}
a_1|x|^3+b_1|x|^2+c_1|x|+d_1, & if\ 0\leqq|x|\leqq 1\\
a_2|x|^3+b_2|x|^2+c_2|x|+d_2, & if\ 1\leqq|x|\leqq 2\\
0, & if\ 2\leqq|x|
\end{cases}
\quad\cdots\cdots(3)
$$

ここで，sinc(x)を近似することから，$\varphi(0)=1$，$\varphi(\pm 1)=0$，$\varphi(\pm 2)=0$とします．そして，これら曲線間を1次微係数まで一致させて滑らかに接続します．したがって，8個の未定係数を解くための連立方程式は式(4)となります．

$$
\begin{cases}
1=\varphi(0)=d_1,\\
0=\varphi(1)=a_1+b_1+c_1+d_1,\\
0=\varphi(1)=a_2+b_2+c_2+d_2,\\
0=\varphi(2)=8a_2+4b_2+2c_2+d_2,\\
0=\varphi'(0)=c_1,\\
\varphi'(1)=3a_1+2b_1+c_1=3a_2+2b_2+c_2,\\
0=\varphi'(2)=12a_2+4b_2+c_2
\end{cases}
\quad\cdots\cdots\cdots\cdots\cdots(4)
$$

連立方程式の数が1個足りず，完全に解けないため，未



(a) キュービック・コンボリューション法　(b) キュービック・スプライン法

図6
キュービック補間法
(a) はキュービック・コンボリューション法の，(b) はキュービック・スプライン法の重み関数ψ(x)．補間の対象となる区間は $x=[0,1]$ にある．そのため，f_{-1} と f_2 の参照点に関する曲線の範囲は，それぞれ $x+1$ と $x-2$ にシフトすることに注意する

定係数が1個(例えばa_2)残ります．そこで，$a_2=a$とします．そうすると，

$$\varphi(x)=\begin{cases}(a+2)|x|^3-(a+3)|x|^3+1, & if\ 0\leqq|x|\leqq1\\ a|x|^3-5a|x|^2+8a|x|-4a, & if\ 1\leqq|x|\leqq2\\ 0, & if\ 2\leqq|x|\end{cases}\quad\cdots\cdots(5)$$

となります．経験的な値として，$a=-1/2$(Keys)，-1(Rifman)などが使われています(ここで，( )内の記述は論文を提出した人の名まえ)．

キュービック・コンボリューション法では，4個の参照点の値f_{-1}，f_0，f_1，f_2を用いて，$\varphi(x)$を重み関数として次のように補間多項式$f(x)$を求めます．

$$f(x)=f_{-1}\varphi(x+1)+f_0\varphi(x-0)+f_1\varphi(x-1)+f_2\varphi(x-2)\quad\cdots\cdots\cdots\cdots\cdots\cdots(6)$$

式(5)を式(6)に代入します．ここで，代入するときにxの範囲が絶対値で定義されているので符号の向きに注意してください．とくに$a=-1/2$のときは式(7)のようになります．

$$\begin{aligned}f(x)=&f_{-1}\left\{-\frac{1}{2}(1+x)^3+\frac{5}{2}(1+x)^2-4(1+x)+2\right\}\\ &+f_0\left(\frac{3}{2}x^3-\frac{5}{2}x^2+1\right)\\ &+f_1\left\{\frac{3}{2}(1-x)^3-\frac{5}{2}(1-x)^2+1\right\}\\ &+f_2\left\{-\frac{1}{2}(2-x)^3+\frac{5}{2}(2-x)^2-4(2-x)+2\right\}\quad\cdots(7)\\ =&\frac{x^3}{2}(3f_0-3f_1-f_{-1}+f_2)\\ &+\frac{x^2}{2}(-5f_0+4f_1+2f_{-1}-f_2)\\ &+\frac{x}{2}(f_1-f_{-1})+f_0\end{aligned}$$

式(7)のような形をした3次多項式を効率良く演算することについては，次回，スプライン法といっしょに詳しく説明します．**図6**に重み関数$\varphi(x)$を示しました．**図6(a)**がキュービック・コンボリューション法です．比較のため，次回で説明するキュービック・スプライン法を**図6(b)**に示しました．

とのむら・もとのぶ
大日本印刷(株) 電子デバイス事業部 電子デバイス研究所

<著者プロフィール>
外村元伸．まだ世の中にパソコンがない時代，大学の大型計算機を使って日食や月食などに関する軌道計算を行い，補間計算法についての知識を身につけました．そして，3次元コンピュータ・グラフィックスのしごとに従事していたときにスプラインと出会いました．それから約20年，画像処理のスプラインで幸運にもアイデアが生まれました．

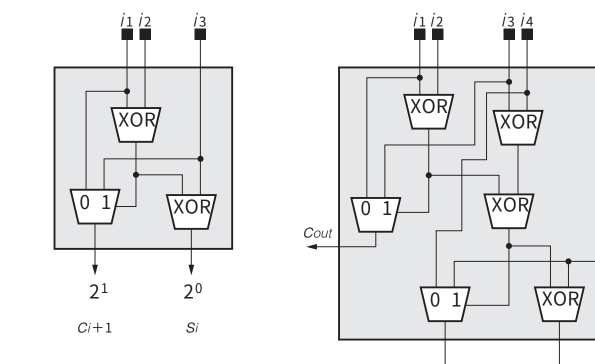
COLUMN：高速カウンタ回路の設計手法

基本カウンタ回路は，3-2カウンタと4-2コンプレッサです(**図A-1**)．線形補間演算回路には，こうしたカウンタ回路を利用します(2006年5月号，pp.39-48の記事に3-2カウンタと2-2カウンタを用いた線形補間演算回路の例を示したので，参照していただきたい)．

ここでは，できるかぎり早い段階でけた上げを出すための組み合わせ回路の設計手法を説明します．7入力のカウンタ回路を例として考えます．$7 = 2^2 + 2^1 + 2^0$なので3ビットの出力となり，それで「7-3カウンタ」と呼んでいます．

図A-2に示すように，7入力を4入力と3入力に分け，4-2コンプレッサと3-2カウンタAの組み合わせで設計します．そして，3-2カウンタAのけたの和を4-2コンプレッサの途中入力端子に入力します．4-2コンプレッサのけた上げ出力2個と3-2カウンタAのけた上げ出力1個で合計3個のけた上げ出力から，さらに1けた上へのけた上げをもう1個の3-2カウンタBで求めます．4-2コンプレッサの途中けた上げ出力と3-2カウンタAのけた上げ出力のタイミングは同じです．また，1タイミング遅れて4-2コンプレッサのもう一つのけた上げ出力が出るので，これらのタイミングを考慮して3-2カウンタBに入力します．このようにして，7入力カウンタの3ビット出力$[b_2 b_1 b_0]$bが得られます．

もっと多入力のカウンタあるいは多数けたで並列カウンタ回路を設計するときに，いちいちゲート回路レベルで記述するのは煩わしいものです．そこで，カウンタ回路の種類，その入出力数とタイミング，そしてけた位置がわかる程度に簡単化して図示する方法として，ドット・ダイアグラム化があります．15-4カウンタを設計するときのドット・ダイアグラム化の方法を**図A-3**に示します．15入力なので，4-2コンプレッサ3個と3-2カウンタの組み合わせが考えられますが，入力できる端子数が余ってしまいます．そこで，7-3カウンタ2個と1入力で組み合わせます．タイミングは，上から下へずらすことで表現します．すでに提案されているドット・ダイアグラムではタイミングを考慮していなかったので，今回はタイミングの概念を入れるくふうを行ってみました．カウンタからの出力線を引き，同じカウンタからの出力の組は線で接続しておきます．けたの位の違いは，色またはラベル付けで表現します．一つのカウンタの入力が散らばってしまうときは，線で接続して表現します．

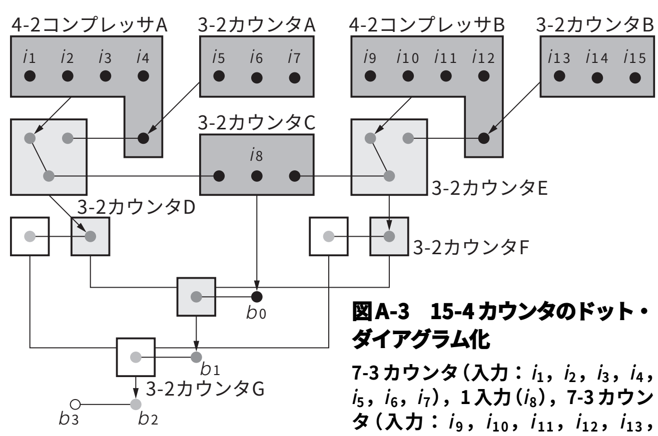


$C_{i+1} = i_1 \wedge \overline{i_1 \oplus i_2} \vee i_3 \wedge (i_1 \oplus i_2)$

$S_i = (i_1 \oplus i_2) \oplus i_3$

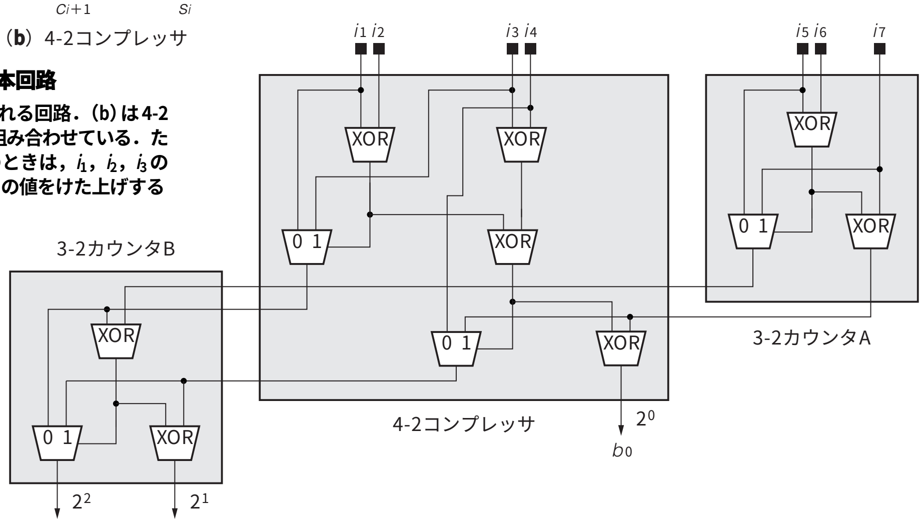
図A-1　カウンタを組み立てるための基本回路

(a)は3-2カウンタでフル・アダーとも呼ばれる回路．(b)は4-2コンプレッサであり，2個の3-2カウンタを組み合わせている．ただし，入力i_1，i_2，i_3，i_4のけたの和が0'のときは，i_1，i_2，i_3のけた上げはc_{out}で出力しているので，残るi_4の値をけた上げする



図A-3　15-4カウンタのドット・ダイアグラム化

7-3カウンタ(入力：i_1，i_2，i_3，i_4，i_5，i_6，i_7)，1入力(i_8)，7-3カウンタ(入力：i_9，i_{10}，i_{11}，i_{12}，i_{13}，i_{14}，i_{15})の組み合わせで設計する．ゲート論理を省き，入出力をドット化して簡略表現する



図A-2
7-3カウンタ

4-2コンプレッサと3-2カウンタAを組み合わせて7入力をカウントする．さらに上位けたへのけた上げをもう一つの3-2カウンタBで求めて，3ビットの出力$[b_2 b_1 b_0]$bを得る